CiteSeer Find: caching hot basic blocks Documents Citations

Searching for PHRASE **caching hot basic blocks**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Amazon B&N Google (CiteSeer) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
1000 documents found. Retrieving documents... **Order: relevance to query.**

Register Pressure Sensitive Redundancy Elimination - Gupta, Bodík (1999) (Correct)
register pressure limits for frequently executed (**hot**) **blocks** and higher limits for infrequently
is the average register pressure y-axis) for all **basic blocks** that have a given execution frequency
pressure limits for frequently executed (**hot**) **blocks** and higher limits for infrequently executed
www.cs.pitt.edu/~gupta/research/Comp/CC99.ps

Profile-Driven Instruction Level Parallel Scheduling with.. - Chekuri Dept (1996) (Correct) (8 citations)
performance in the face of fixed instruction **cache** sizes. In light of this, the threshold and the
application of a scheduling heuristic is limited to **basic blocks**, considerable performance loss may be
Parallel Scheduling with Application to Super **Blocks** C. Chekuri Dept. of Comp. Sci. Stanford Univ.
theory.stanford.edu/~chekuri/postscript/micro96.ps.gz

Modeling Caching Effect in Continuous Media Server - Kang, Yeom (1999) (Correct)
Modeling **Caching** Effect in Continuous Media Server Sooyong Kang
dcslab.snu.ac.kr/~yeom/paper/mascots99.ps

Motivation - Subprogram Inlining (Correct)
calculator dinero, a trace-driven **cache** simulator developed at the University of
function, used in many scientific programs, has a **hot** spot that covers only a third of its code,
that covers only a third of its code, spanning two **basic blocks** and 69 instructions out of a total of 14
www.cs.arizona.edu/people/debray/papers/partial-inlining.ps

Predicting Worst Case Execution Times on a Pipelined RISC.. - Bharrat, Jeffay (1995) (Correct) (3 citations)
Modern computer systems with pipelined processors, **caches**, DMA, etc.can complicate this process. We
the computation of worst case execution times for **basic blocks** -the lowest level and most processor
of worst case execution times for **basic blocks** -the lowest level and most processor dependent
ftp.cs.unc.edu/pub/users/jeffay/papers/Bharrat.ps.Z

The Execution Order Control Method among Coarse Grain.. - Koichi Asakura (Correct)
order control algorithms are designed for the **basic-block**-oriented distributed processes. However, we
order control algorithms are designed for the **basic-block**-oriented distributed processes. However, we
for the distributed processes in terms of **basic blocks**. Therefore, even if these methods were adopted to
fcapwide.fujitsu.co.jp/pcw/pcw95j/p2c.ps.gz

Multiscalar Processors - Sohi (1995) (Correct) (165 citations)
instructions can be maintained in the instruction **cache**, so that the overhead of accessing two memory
are also presented. 1. Introduction The **basic** paradigm of sequencing through a program, i.e.
program as a control flow graph (CFG)where **basic blocks** are nodes, and arcs represent flow of control
davinci.snu.ac.kr/links/ilp/sohi95.ps.gz

Efficient Cooperative Caching using Hints - Sarkar, Hartman (1996) (Correct) (31 citations)
Efficient Cooperative **Caching** using Hints Prasenjit Sarkar and John Hartman
www.cs.arizona.edu/swarm/papers/ccache/paper.ps

Global Register Allocation Based on Graph Fusion - Guei-Yuan Lueh (1996) (Correct) (7 citations)
eachregion of the program, where a region can be a **basic block**, a loop nest, a superblock, a trace, or
of the program, where a region can be a **basic block**, a loop nest, a superblock, a trace, or another
a trace, or another combination of **basic blocks**. Region formation is orthogonal to register
www.cs.cmu.edu/afs/cs.cmu.edu/project/iwarp/archive/fx-papers/lcpc96.ps

The SUIF Control Flow Graph Library - Young (1998) (Correct) (1 citation)
affect performance due to negative branch and **cache** effects. Optimizations such as code layout [2]
an abstraction of control flow graphs built on the **basic** structures of the SUIF system. The CFG library
transforming them by rearranging and reconnecting **blocks**, and fine-grained control over individual program
www.eecs.harvard.edu/machsuif/machsuif-doc/cfg.ps

Near-Optimal Parallel Prefetching and Caching - Kimbrel, Karlin (1997) (Correct) (28 citations)
Near-optimal parallel prefetching and **caching** Tracy Kimbrel y Anna R. Karlin z August 29,
www.cs.washington.edu/homes/tracyk/focs-long.ps

On Caching Search Engine Results - Markatos (1999) (Correct) (7 citations)
On **Caching** Search Engine Results Evangelos P. Markatos
www.ccsf.caltech.edu/~markatos/avg/papers/1999.TR241.Caching_search_engines.ps.gz

Resource-based Caching for Web Servers - Renu Tewari (1998) (Correct) (35 citations)
Resource-based **Caching** for Web Servers Renu Tewari, Harrick M. Vin,
www.cs.utexas.edu/users/dmci/papers/ps/MMCN98-RBC.ps

Application-Controlled File Caching Policies - Cao, Felten, Li (1994) (Correct) (52 citations)
Application-Controlled File **Caching** Policies Pei Cao, Edward W. Felten, and Kai Li
allows processes to manage their own **cache blocks**, while at the same time maintains the dynamic
time maintains the dynamic allocation of **cache blocks** among processes. Our solution makes sure that
ftp.cs.princeton.edu/reports/1994/445.ps.Z

The Machine SUIF Bit-Vector Data-Flow-Analysis Library - Holloway (1998) (Correct)
[4] to parse the program being analyzed into **basic blocks**, and it associates data-flow results with
[4] to parse the program being analyzed into **basic blocks**, and it associates data-flow results with these
them monotonically to reflect the effects of **basic blocks** and the confluence of edges, until they converge
www.eecs.harvard.edu/~hube/machsuif2-doc/bvd.ps

Instruction Cache Effects of Different Code Reordering Algorithms - Lee (1994) (Correct) (4 citations)
Instruction **Cache** Effects of Different Code Reordering Algorithms
www.cs.washington.edu/homes/dlee/mypapers/quals.ps

The Effect of Client Caching on File Server Workloads - Kevin Froese (1996) (Correct) (6 citations)
The Effect of Client **Caching** on File Server Workloads Kevin W. Froese
www.cs.usask.ca/staff/kwf230/research/hicss96.ps.gz

Resource Spackling: A Framework for Integrating Register... - Berson, Gupta, Soffa (1994) (Correct) (16 citations)
scheduling algorithm that moves instructions across **basic block** boundaries only when resource holes exist
algorithm that moves instructions across **basic block** boundaries only when resource holes exist and
scheduler schedules instructions within a **basic block**, while a global scheduler exploits interblock
ftp.cs.pitt.edu/berson/papers/tr94-09.ps

On the Limit of Control Flow Analysis for Regression Test Selection - Ball (1998) (Correct) (12 citations)
P's control flow graph G, each vertex represents a **basic block** of instructions and each edge represents a
flow graph G, each vertex represents a **basic block** of instructions and each edge represents a
each edge represents a control transition between **blocks**. The translation of an abstract syntax tree
www.bell-labs.com/~tball/papers/issta98.ps.gz

Global Instruction Scheduling In Machine SUIF - Gang Chen (1997) (Correct) (2 citations)
no matter whether we schedule locally (within a **basic block**) or globally (across **basic blocks**)For
matter whether we schedule locally (within a **basic block**) or globally (across **basic blocks**)For global
(within a **basic block**) or globally (across **basic blocks**)For global instruction scheduling, since
www.eecs.harvard.edu/machsuif/papers/hpca3.ps

First 20 documents Next 20

Try your query at: Amazon Barnes & Noble Google (CiteSeer) Google (Web) CSB DBLP

CiteSeer Find: caching most frequently executed bas Documents Citations

Searching for PHRASE **caching most frequently executed basic blocks**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Amazon B&N Google (CiteSeer) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
1000 documents found. **Only retrieving 500 documents (System busy - maximum reduced).** Retrieving documents...
Order: relevance to query.

Efficient Path Profiling - Ball, Larus (1996) (Correct) (62 citations)
such as the number of processor cycles, stalls, **cache** misses, or page faults. A minor change to the
shows that the SPEC95 train input datasets covered **most** of the paths **executed** in the ref datasets. This
which edge profiling does not identify the **most frequently executed** paths. The table contains two
www.stanford.edu/class/cs343/ps/pathprof.ps

Reducing Branch Costs via Branch Alignment - Calder, Grunwald (1994) (Correct) (32 citations)
these algorithms has been on improving instruction **cache** locality, and the few studies concerned with
(ASPLOS-VI)San Jose, California. October 1994. **most** recent address. If the decoded instruction breaks
graph so that fall-through branches occur more **frequently**. We use profile information to direct the
www.cs.colorado.edu/~grunwald/GCAG/dirk-arch-asplos94.ps

Profile-Driven Instruction Level Parallel Scheduling with.. - Chekuri Dept (1996) (Correct) (8 citations)
performance in the face of fixed instruction **cache** sizes. In light of this, the threshold and the
scheme is computationally intractable in the **most** general case, it is practicable for super **blocks**
the branches via hardware support for predicated **execution**, allowing instructions to be moved outside of
theory.stanford.edu/~chekuri/postscript/micro96.ps.gz

Instruction Cache Effects of Different Code Reordering Algorithms - Lee (1994) (Correct) (4 citations)
Instruction **Cache** Effects of Different Code Reordering Algorithms
www.cs.washington.edu/homes/diee/mypapers/quals.ps

Path Profile Guided Partial Redundancy Elimination Using.. - Gupta, Berson, Fang (1997) (Correct) (9 citations)
we can see the number of functions that require at **most** 5 paths increases substantially (from 1694 to
be designed to trade off the performance of less **frequently executed** paths in favor of more **frequently**
these paths are typically exercised during program **execution**. Thus, optimization algorithms should be
www.cs.pitt.edu/~gupta/research/Comp/icci98b.ps

Register Pressure Sensitive Redundancy Elimination - Gupta, Bodík (1999) (Correct)
when not enough registers are available, the **most** profitable redundancies are removed first. To
By setting strict register pressure limits for **frequently executed** (hot) **blocks** and higher limits for
strict register pressure limits for **frequently executed** (hot) **blocks** and higher limits for infrequently
www.cs.pitt.edu/~gupta/research/Comp/CC99.ps

Software Trace Cache - Ramírez, Larriba-Pey.. (1999) (Correct) (2 citations)
Software Trace **Cache** Alex Ram'irez Josep-L. Larriba-Pey Carlos Navarro
ftp.ac.upc.es/pub/reports/DAC/1999/UPC-DAC-1999-5.ps.Z

Global Register Allocation Based on Graph Fusion - Guei-Yuan Lueh (1996) (Correct) (7 citations)
compiler to pick the heuristic or strategy that is **most** in line with the rest of the compiler design. 2
compiler design. 2 Background and prior work A **frequently** employed technique is to first allocate a
by a Chatin-style allocator. This algorithm uses **execution** probabilities, derived from either profiles or
www.cs.cmu.edu/afs/cs.cmu.edu/project/iwarp/archive/fx-papers/lcpc96.ps

Initial Results for Glacial Variable Analysis - Tito Autrey (1996) (Correct) (17 citations)
excellent candidate variables. 5 Related Work The **most** closely related work to glacial variable analysis
candidate variables. They are modified much less **frequently** than they are referenced. In current systems
the total run-time of the program is reduced. The **execution** time savings must exceed the cost of RTCG.
www.cse.ogi.edu/Sparse/paper/glacial.lcpc.96.ps

The Execution Order Control Method among Coarse Grain.. - Koichi Asakura (Correct)
The **Execution** Order Control Method among Coarse Grain
order control algorithms are designed for the **basic-block**-oriented distributed processes. However, we
order control algorithms are designed for the **basic-block**-oriented distributed processes. However, we

fcapwide.fujitsu.co.jp/pcw/pcw95j/p2c.ps.gz

Multiscalar Processors - Sohi (1995) (Correct) (165 citations)
instructions can be maintained in the instruction **cache**, so that the overhead of accessing two memory a multiscalar processor is shown in Figure 1. In **most** general terms, consider a multiscalar processor to complex. Each of these units fetches and **executes** instructions belonging to its assigned task. The
davinci.snu.ac.kr/links/ilp/sohi95.ps.gz

Experiments with Data Flow and Mutation Testing - Offutt, Pan, Zhang, Tewary (1994) (Correct) (1 citation)
detects all simple faults in a program will detect **most** complex faults. Simple faults are introduced into other. Second, we compare mutation and all-uses by **executing** faulty versions of programs and comparing how function)A subprogram is decomposed into a set of **basic blocks**, which are maximal sequences of simple
www.isse.gmu.edu/techrep/1994/94_105_offutt.ps

Comparing Static and Dynamic Scheduling on Superscalar Processors - Lo (1995) (Correct)
techniques may be insufficient. Non-**blocking caches**[Kro81]FJ94] expose latencies that are difficult and code scheduling was pushed into software. **Most** modern compilers provide instruction scheduling as through the program will be **executed** more **frequently** than others. Furthermore, the compiler must be
www.cs.washington.edu/homes/jlo/papers/generals.ps

Predicting Worst Case Execution Times on a Pipelined RISC.. - Bharrat, Jeffay (1995) (Correct) (3 citations)
Modern computer systems with pipelined processors, **caches**, DMA, etc.can complicate this process. We worst case **execution** times a difficult problem. **Most** programs are written in a higher level language, Predicting Worst Case **Execution** Times on a Pipelined RISC Processor Shaun J.
ftp.cs.unc.edu/pub/users/jeffay/papers/Bharrat.ps.Z

Modeling Caching Effect in Continuous Media Server - Kang, Yeom (1999) (Correct)
Modeling **Caching** Effect in Continuous Media Server Sooyong Kang
doslab.snu.ac.kr/~yeom/paper/mascots99.ps

On Caching Search Engine Results - Markatos (1999) (Correct) (7 citations)
On **Caching** Search Engine Results Evangelos P. Markatos
www.ccsf.caltech.edu/~markatos/avg/papers/1999.TR241.Caching_search_engines.ps.gz

Using Profile Information to Assist Classic Code Optimizations - Chang (1991) (Correct) (53 citations)
W. Hwu and P. P. Chang, Achieving High Instruction **Cache** Performance with an Optimizing Compiler" profiling tools allow programmers to identify the **most** important functions and the **most frequently** the **execution** time by moving instructions from **frequently executed** program regions to infrequently
ftp.crhc.uiuc.edu/pub/IMPACT/journal/spe.profile-classic.91.ps

Some MPEG Decoding Functions on Spert An Example for Assembly.. - Formella (1994) (Correct) (1 citation)
index addresses and all instructions reside in the **cache** we calculate the upper bound of the run time as of Assembly Program 23 1 Introduction The **most** recent documentation (or at least pointer to that operation needs at **most** 4 clock cycle to be **executed**. ffl chaining of operations is possible ffl
ftp.icsi.berkeley.edu/pub/techreports/1994/tr-94-027.ps.gz

Register Allocation for Predicated Code - Eichenberger, Davidson (1995) (Correct) (5 citations)
predicate values, an additional source operand for **most** operations to spec183 Code fragments Intermediate [2] used in the IMPACT compiler combines **frequently executed basic blocks** from multiple **execution** Keywords: Register Allocation, Predicated **Execution**, Interference, Hyperblocks, Software
ftp.eecs.umich.edu/groups/PPP/BAK-OLD/MICRO95b.ps

First 20 documents Next 20

Try your query at: Amazon Barnes & Noble Google (CiteSeer) Google (Web) CSB DBLP

CiteSeer Find: caching most frequently executed bas Documents Citations

Searching for PHRASE **caching most frequently executed basic blocks**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Amazon B&N Google (CiteSeer) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
1000 documents found. Retrieving documents... **Order: relevance to query.**

Synthesis Of Power Efficient Systems-On-Silicon - Kirovski, Lee, Potkonjak.. (1998) (Correct) (3 citations)
in three steps: minimization of instruction **cache** misses, placement of **frequently executed**
www.cs.ucla.edu/~darko/papers/aspdac98.ps.gz

Efficient Organization of Control Structures in Distributed.. - Hogen, Loogen (Correct)
Nevertheless, their approach yields a good **cache** behaviour, which should also be observed in our
In purely sequential implementations of **most** programming languages a runtime stack is used for
stacks of several parallel processes, which are **executed** on the same processor element, are stored in an
www-i2.informatik.rwth-aachen.de/OldStaff/hogen/PUBLICATIONS/cc94.ps.gz

Application-Controlled File Caching Policies - Cao, Felten, Li (1994) (Correct) (52 citations)
Application-Controlled File **Caching** Policies Pei Cao, Edward W. Felten, and Kai Li
Policy The kernel allocation policy is the **most** critical part of two-level replacement. To obtain
ftp.cs.princeton.edu/reports/1994/445.ps.Z

Efficient Cooperative Caching using Hints - Sarkar, Hartman (1996) (Correct) (31 citations)
Efficient Cooperative **Caching** using Hints Prasenjit Sarkar and John Hartman
www.cs.arizona.edu/swarm/papers/ccache/paper.ps

A Quantitative Analysis of Loop Nest Locality - McKinley, Temam (1996) (Correct) (27 citations)
future directions for architecture and software **cache** optimizations. Since **most** programs spend the
www.masi.uvsq.fr/~temam/Articles/McTe96.ps.gz

The SUIF Control Flow Graph Library - Young (1998) (Correct) (1 citation)
affect performance due to negative branch and **cache** effects. Optimizations such as code layout [2]
feel that this is not a huge restriction, because **most** layout or code motion optimizations require
implementation assumptions. ffl Continuous **Execut**ability. As a final design goal, we wanted the
www.eecs.harvard.edu/machsuif/machsuif-doc/cfg.ps

Digital Equipment Corporation Hudson, Massachusetts - Us Et Ts (Correct)
Called Nt Om, That Arranges Code For Instruction **Cache** Performance Using Profile Information, Nt
collection of large Windows NT applications. HCO is **most** effective on the programs that are call intensive
information to partition each routine into **frequently executed** (hot) and infrequently **executed** (cold)
ftp.digital.fr/pub/DEC/Micro29.ps

Resource Spackling: A Framework for Integrating Register.. - Berson, Gupta, Soffa (1994) (Correct) (16 citations)
a decision must be made as to which set is **most** beneficial to move. Only sets which the
sets, which are sets of instructions that may be **executed** concurrently but require more resources than are
scheduling algorithm that moves instructions across **basic block** boundaries only when resource holes exist
ftp.cs.pitt.edu/berson/papers/tr94-09.ps

On the Limit of Control Flow Analysis for Regression Test Selection - Ball (1998) (Correct) (12 citations)
The above definition translates trivially into the **most** precise and computationally expensive CRTS
(such as coverage information) collected about the **execut**ion old(t) in order to make this determination. An
P's control flow graph G, each vertex represents a **basic block** of instructions and each edge represents a
www.bell-labs.com/~tball/papers/issta98.ps.gz

Near-Optimal Parallel Prefetching and Caching - Kimbrel, Karlin (1997) (Correct) (28 citations)
Near-optimal parallel prefetching and **caching** Tracy Kimbrel y Anna R. Karlin z August 29,
www.cs.washington.edu/homes/tracyk/focs-long.ps

Global Instruction Scheduling In Machine SUIF - Gang Chen (1997) (Correct) (2 citations)
are ready to begin scheduling. Our algorithm, like **most** current algorithms, schedules the region one
of programs, even those without strongly biased **execut**ion paths, and both employ heuristics to avoid
no matter whether we schedule locally (within a **basic block**) or globally (across **basic blocks**)For

www.eecs.harvard.edu/machsuif/papers/hpca3.ps

Using Profile Information to Assist Advanced.. - Chen, Mahlke.. (1992) (Correct) (4 citations)
the program **execution** time, assuming a 100% **cache** hit rate, is reported as a speedup relative to profile information, the compiler can identify the **most frequently** invoked calls and determine the best the compiler must be able to identify the **frequently executed** sequences of **basic blocks** in a flow
ftp.crhc.uiuc.edu/pub/IMPACT/book/advances.profile.93.ps

Optimizing Instruction Cache Performance for Operating.. - Torrellas, Xia, Daigle (1995) (Correct) (27 citations)
Optimizing Instruction **Cache** Performance for Operating System Intensive
polaris.cs.uiuc.edu/reports/1387.ps.gz

Application-Driven Synthesis of Core-Based Systems - Ms (Correct)
guide the algorithm for minimization of instruction **cache** misses for a given application, instruction **cache**
ftp.cs.ucla.edu/tech-report/97-reports/970028.ps.Z

The Machine SUIF Bit-Vector Data-Flow-Analysis Library - Holloway (1998) (Correct)
7i 2.1 Accessing data-flow results Note that **most** methods of bvd are protected from public use.
[4] to parse the program being analyzed into **basic blocks**, and it associates data-flow results with
[4] to parse the program being analyzed into **basic blocks**, and it associates data-flow results with these
www.eecs.harvard.edu/~hube/machsuif2-doc/bvd.ps

Better Global Scheduling Using Path Profiles - Cliff Young (1998) (Correct) (2 citations)
of this work on hardware techniques such as trace **caches** [13,16]2 Superblock formation A superblock exits before the end of the trace, since the **most** aggressive compaction algorithms aim to minimize of **basic blocks** that only approximate the **frequently-executed** program paths. The identified
www.eecs.harvard.edu/hube/papers/micro98-superpath.ps

Profile-Guided Context-Sensitive Program Analysis - Debray (Correct) (2 citations)
For example, in the SPEC-95 benchmark m88ksim, the **most frequently** called function, uext(has 19 call tends to be skewed towards a small number of **frequently executed** call sites. For example, in the procedure, and back to the **basic block** to which **execution** returns at the end of the call. Traditionally,
www.cs.arizona.edu/people/debray/papers/pgcsens.ps

Memory Behavior of the SPEC2000 Benchmark Suite - Sair, Charney (Correct)
In this paper we present measurements of number of **cache** misses for all the applications for a variety of
www.cs.ucsd.edu/~ssair/rc21852.ps

The Effect of Client Caching on File Server Workloads - Kevin Froese (1996) (Correct) (6 citations)
The Effect of Client **Caching** on File Server Workloads Kevin W. Froese
www.cs.usask.ca/staff/kwf230/research/hicss96.ps.gz

Documents 21 to 40 Previous 20 Next 20

Try your query at: Amazon Barnes & Noble Google (CiteSeer) Google (Web) CSB DBLP

CiteSeer Find: caching hot block Documents Citations

Searching for PHRASE **caching hot blocks**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Amazon B&N Google (CiteSeer) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
1000 documents found. Retrieving documents... **Order: relevance to query.**

Hot Block Clustering for Disk Arrays - With Dynamic Striping (1995) (Correct)
unit[2, 3]floating parity/data[6]smart **caching**[5]parity logging[11]LRAID[1] and dynamic **Hot Block** Clustering for Disk Arrays with Dynamic
www.tkl.iis.u-tokyo.ac.jp/Kilab/Research/Paper/./1995/mogi/HBC.ps

Modeling Caching Effect in Continuous Media Server - Kang, Yeom (1999) (Correct)
Modeling **Caching** Effect in Continuous Media Server Sooyong Kang
dcslab.snu.ac.kr/~yeom/paper/mascots99.ps

Register Pressure Sensitive Redundancy Elimination - Gupta, Bodík (1999) (Correct)
register pressure limits for frequently executed (**hot**) **blocks** and higher limits for infrequently pressure limits for frequently executed (**hot**) **blocks** and higher limits for infrequently executed and higher limits for infrequently executed (cold) **blocks**, our algorithm permits trade-off between
www.cs.pitt.edu/~gupta/research/Comp/CC99.ps

Motivation - Subprogram Inlining (Correct)
calculator dinero, a trace-driven **cache** simulator developed at the University of function, used in many scientific programs, has a **hot** spot that covers only a third of its code, only a third of its code, spanning two basic **blocks** and 69 instructions out of a total of 14 basic
www.cs.arizona.edu/people/debray/papers/partial-inlining.ps

Efficient Cooperative Caching using Hints - Sarkar, Hartman (1996) (Correct) (31 citations)
Efficient Cooperative **Caching** using Hints Prasenjit Sarkar and John Hartman
www.cs.arizona.edu/swarm/papers/ccache/paper.ps

Near-Optimal Parallel Prefetching and Caching - Kimbrel, Karlin (1997) (Correct) (28 citations)
Near-optimal parallel prefetching and **caching** Tracy Kimbrel y Anna R. Karlin z August 29,
www.cs.washington.edu/homes/tracyk/focs-long.ps

On Caching Search Engine Results - Markatos (1999) (Correct) (7 citations)
On **Caching** Search Engine Results Evangelos P. Markatos
www.ccsf.caltech.edu/~markatos/avg/papers/1999.TR241.Caching_search_engines.ps.gz

Resource-based Caching for Web Servers - Renu Tewari (1998) (Correct) (35 citations)
Resource-based **Caching** for Web Servers Renu Tewari, Harrick M. Vin,
www.cs.utexas.edu/users/dmcl/papers/ps/MMCN98-RBC.ps

Application-Controlled File Caching Policies - Cao, Felten, Li (1994) (Correct) (52 citations)
Application-Controlled File **Caching** Policies Pei Cao, Edward W. Felten, and Kai Li
allows processes to manage their own **cache blocks**, while at the same time maintains the dynamic time maintains the dynamic allocation of **cache blocks** among processes. Our solution makes sure that
ftp.cs.princeton.edu/reports/1994/445.ps.Z

Adaptive Block Rearrangement Under UNIX - Akyurek, Salem (1994) (Correct) (3 citations)
data **blocks**. The operating system also manages the **caching** of file **blocks** in main memory buffers. All file that try to cluster the **blocks** of a file. However, **hot blocks** from different files may be spread widely
UMIACS-TR-93-28.1 February, 1994 Adaptive **Block** Rearrangement Under UNIX Sedat Akyurek
zonker.uwaterloo.ca/pub/TRs/3054.1.ps.Z

The Effect of Client Caching on File Server Workloads - Kevin Froese (1996) (Correct) (6 citations)
The Effect of Client **Caching** on File Server Workloads Kevin W. Froese
www.cs.usask.ca/staff/kwf230/research/hicss96.ps.gz

On Performance of Caching Proxies - Rousskov (Correct) (41 citations)
On Performance of **Caching** Proxies Alex Rousskov Valery Soloviev

www.cs.ndsu.nodak.edu/~rousskov/research/cache/squid/profiling/papers/on.performance.ps.gz

A Performance Study of the Squid Proxy on HTTP/1.0 - Rousskov, Soloviev (1999) (Correct) (3 citations)
a performance study of the state-of-the-art **caching** proxy called Squid. We instrumented Squid to
www.cs.ndsu.nodak.edu/~rousskov/research/cache/squid/profiling/papers/wwwj99.ps.gz

Cooperative Caching for Financial Databases with Hot Spots - Aman Sinha (1998) (Correct)
Disciplina Praesidium Civitatis Cooperative **Caching** For Financial Databases With **Hot** Spots Aman
maple.ece.utexas.edu/TechReports/1998/TR-PDS-1998-009.ps.Z

Cache Investment Strategies - Franklin, Kossmann (1997) (Correct) (1 citation)
Cache Investment Strategies Michael J. Franklin
www.cs.umd.edu/projects/dimsum/papers/cinvest.ps.gz

Continuous Multicast Push of Web Documents over the Internet - Rodriguez, Biersack (1998) (Correct) (3 citations)
that change very frequently and that are not worth **caching**. A Web server using CMP continuously multicasts traffic on the Internet. Popular Web pages create "**hot** spots" of network load due to their great demand
www.eurecom.fr/~btroup/BPublished/RODRI98_cmp.ps.gz

Cache-Conscious Structure Definition - Chilimbi, Davidson, Larus (1999) (Correct) (18 citations)
and Implementation, May 1999. ABSTRACT A program's **cache** performance can be improved by changing the
ftp.cs.wisc.edu/wwt/pldi99_cache_def.ps

Video-on-Demand on the SB-PRAM - Friedrich, Grün, Keller (1996) (Correct) (2 citations)
few processors like the Sequent Symmetry [2] or are **cache**-based like the KSR1/2 [13] or the Stanford DASH hashed among the memory modules, thus avoiding **hot** spots [8]The bandwidth between processors and allows for a very regular distribution of **blocks** onto disks, but restricts bit rates to few
www-wjp.cs.uni-sb.de/~jf/nossdav.ps.gz

Proxy Caching Mechanism for Multimedia Playback Streams .. - Rejaie, Handley, Yu.. (1999) (Correct) (14 citations)
Proxy **Caching** Mechanism for Multimedia Playback Streams in the
netweb.usc.edu/reza/papers/mc.ps

Client-Caching Algorithms in a Video-on-Demand System - Defeng Ma (Correct)
Client-**Caching** Algorithms in a Video-on-Demand System Defeng
www.inf.ethz.ch/personal/alonso/PAPERS/cintcache.ps.Z

First 20 documents Next 20

Try your query at: Amazon Barnes & Noble Google (CiteSeer) Google (Web) CSB DBLP

CiteSeer Find: caching hot spots Documents Citations

Searching for PHRASE **caching hot spots**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Amazon B&N Google (CiteSeer) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
1000 documents found. Retrieving documents... **Order: relevance to query.**

Consistent Hashing and Random Trees: Distributed.. - Karger, Lehman.. (1997) (Correct) (92 citations)
Consistent Hashing and Random Trees: Distributed **Caching** Protocols for Relieving **Hot Spots** on the World
theory.lcs.mit.edu/~karger/Papers/web.ps.gz

Cooperative Caching for Financial Databases with Hot Spots - Aman Sinha (1998) (Correct)
Disciplina Praesidium Civitatis Cooperative **Caching** For Financial Databases With **Hot Spots** Aman
maple.ece.utexas.edu/TechReports/1998/TR-PDS-1998-009.ps.Z

Mining the Knowledge Mine: The Hot Spots Methodology for.. - Williams, Huang (1997) (Correct) (2 citations)
Mining the Knowledge Mine The **Hot Spots** Methodology for Mining Large Real World
Mining the Knowledge Mine The **Hot Spots** Methodology for Mining Large Real World
Mining the Knowledge Mine: The **Hot Spots** Methodology for Mining Large Real World
www.cmis.csiro.au/Graham.Williams/papers/ai97.ps.gz

A Distributed Directory Cache Coherence Scheme and its Effects.. - Gupta, al. (1995) (Correct) (1 citation)
High Performance Computing A Distributed Directory **Cache** Coherence Scheme and its Effects on Network
yara.ecn.purdue.edu/~abraham/papers/jhpc95.ps.Z

An Effective Synchronization Network for Hot-spot Accesses - Hsu, Yew (1992) (Correct) (4 citations)
fetch&tail,1) mod qsize*qsize/qsize assumed **cache**d *3: wait for response if (mytail 1) mod
An Effective Synchronization Network for **Hot-spot** Accesses William Tsun-yuk Hsu and Pen-chung
An Effective Synchronization Network for **Hot-spot** Accesses William Tsun-yuk Hsu and Pen-chung Yew
www.csrd.uiuc.edu/reports/952.ps.gz

CCHIME: A Cache Coherent Hybrid Interconnected Memory.. - Farrens, Park, Woodruff (Correct)
CCHIME: A **Cache** Coherent Hybrid Interconnected Memory Extension
american.cs.ucdavis.edu/publications/IPPS.92.ps

Space-Efficient Hot Spot Estimation - Kenneth Salem (1993) (Correct)
and end of the scan, respectively. 2.1 The Name **Cache** Algorithm The Name **Cache** (NC) algorithm
Space-Efficient **Hot Spot** Estimation Kenneth Salem Institute for
ftp.cs.umd.edu/pub/papers/papers/ncstri.umcp/CS-TR-3115/CS-TR-3115.ps.Z

On the "Hot Spots" Conjecture of J. Rauch - Bañuelos, Burdzy (Correct)
On The "**hot Spots**" Conjecture Of J. Rauch Rodrigo Ba~nuelos*
x 2 @D t ?0: 1:1) Informally speaking, the "**hot spots**" conjecture of J. Rauch asserts that, in the
D as t goes to infinity. In other words, the "**hot spots**" move towards the boundary. We will state several
www.math.washington.edu/~burdzy/Papers/hotspot.ps

Array Combining Scatter Functions on Coarse-Grained.. - Bae, Alsabti, Ranka (1997) (Correct) (2 citations)
for array combining scatter functions with arbitrary **hot spots** (processors) on coarse-grained,
combining scatter functions with arbitrary **hot spots** (processors) on coarse-grained,
high performance fortran, random access write, **hot spot**, **hot** processor, direct algorithm, two-stage
rose1.etri.re.kr/~sbae/PS-File/crpc98.ps.gz

Alleviation of Tree Saturation in Multistage.. - Farrens, Wetmore.. (1991) (Correct) (3 citations)
came out in bursts. Further, the effects of **caching** have yet to be examined, although Pfister and
but complementary extensions of previous work on **hot spot** contention in multistage interconnection
complementary extensions of previous work on **hot spot** contention in multistage interconnection
american.cs.ucdavis.edu/publications/Supercomputing91.ps

Continuous Multicast Push of Web Documents over the Internet - Rodriguez, Biersack (1998) (Correct) (3 citations)
that change very frequently and that are not worth **caching**. A Web server using CMP continuously multicasts
traffic on the Internet. Popular Web pages create "**hot spots**" of network load due to their great demand
www.eurecom.fr/~btroup/BPublished/RODRI98_cmp.ps.gz

Motivation - Subprogram Inlining (Correct)
calculator dinero, a trace-driven **cache** simulator developed at the University of
function, used in many scientific programs, has a **hot spot** that covers only a third of its code,
used in many scientific programs, has a **hot spot** that covers only a third of its code, spanning two
www.cs.arizona.edu/people/debray/papers/partial-inlining.ps

Resource-based Caching for Web Servers - Renu Tewari (1998) (Correct) (35 citations)
Resource-based **Caching** for Web Servers Renu Tewari, Harrick M. Vin,
www.cs.utexas.edu/users/dmci/papers/ps/MMCN98-RBC.ps

A Hierarchical Internet Object Cache - Chankhunthod, Danzig, Neerdaels.. (1995) (Correct) (270 citations)
A Hierarchical Internet Object **Cache** Anawat Chankhunthod Peter B. Danzig Chuck
ftp.cs.colorado.edu/pub/techreports/schwartz/HarvestCache.ps.Z

Effect of Non-uniform Traffic on the Performance of.. - Atiquzzaman Akhtar (1993) (Correct) (1 citation)
to memories in shared memory multiprocessor systems. **Hot spots** in shared memory multiprocessor systems
in shared memory multiprocessor systems. **Hot spots** in shared memory multiprocessor systems result in
between unbuffered and buffered MINs under **hot spot** traffic pattern has been presented in this paper.
www.engr.udayton.edu/faculty/matiquzz/papers/om-hot4.ps

Credit-Flow-Controlled ATM versus Wormhole Routing - Katevenis, Serpanos, Spyridakis (1996) (Correct)
wordisalready one thirdofanATM cell in size in **cache**-coherent multiprocessors, small packets
unlike wormhole, it is fair in terms of latency in **hot-spot** configurations. Our simulation uses detailed
www.ii.uib.no/~markatos/arch-vlsi/papers/1996.TR171.ATM_vs_Wormhole.ps.gz

Cooperative Caching in Append-only Databases with Hot Spots - Aman Sinha (Correct)
Cooperative **Caching** in Append-only Databases with **Hot Spots** Aman
Cooperative **Caching** in Append-only Databases with **Hot Spots** Aman Sinha and Craig Chase Parallel and
lore.ece.utexas.edu/~sinha/ICDE.ps

Proxy Caching Mechanism for Multimedia Playback Streams .. - Rejaie, Handley, Yu.. (1999) (Correct) (14 citations)
Proxy **Caching** Mechanism for Multimedia Playback Streams in the
netweb.usc.edu/reza/papers/mc.ps

Client-Caching Algorithms in a Video-on-Demand System - Defeng Ma (Correct)
Client-**Caching** Algorithms in a Video-on-Demand System Defeng
www.inf.ethz.ch/personal/alonso/PAPERS/cintcache.ps.Z

Random Data Accesses on a Coarse-grained Parallel Machine II.. - Ravi Shankar (1997) (Correct) (6 citations)
algorithms for performing random data accesses with **hot spots** on a coarse-grained parallel machine. The
for performing random data accesses with **hot spots** on a coarse-grained parallel machine. The general
random access read/write operations with **hot spots** can be completed in Cn=p (lower order terms)
www.npac.syr.edu/projects/pcrc/doc/florida/RandomDataAccess2.ps

First 20 documents Next 20

Try your query at: Amazon Barnes & Noble Google (CiteSeer) Google (Web) CSB DBLP

CiteSeer Find: caching most frequently executed coc Documents Citations

Searching for PHRASE **caching most frequently executed code blocks**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Amazon B&N Google (CiteSeer) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
1000 documents found. Retrieving documents... **Order: relevance to query.**

Efficient Path Profiling - Ball, Larus (1996) (Correct) (62 citations)
such as the number of processor cycles, stalls, **cache** misses, or page faults. A minor change to the shows that the SPEC95 train input datasets covered **most** of the paths **executed** in the ref datasets. This which edge profiling does not identify the **most frequently executed** paths. The table contains two
www.stanford.edu/class/cs343/ps/pathprof.ps

Path Profile Guided Partial Redundancy Elimination Using.. - Gupta, Berson, Fang (1997) (Correct) (9 citations)
we can see the number of functions that require at **most** 5 paths increases substantially (from 1694 to be designed to trade off the performance of less **frequently executed** paths in favor of more **frequently** these paths are typically exercised during program **execution**. Thus, optimization algorithms should be
www.cs.pitt.edu/~gupta/research/Comp/iccl98b.ps

Instruction Cache Effects of Different Code Reordering Algorithms - Lee (1994) (Correct) (4 citations)
Instruction **Cache** Effects of Different **Code** Reordering Algorithms
www.cs.washington.edu/homes/dlee/mypapers/quals.ps

Reducing Branch Costs via Branch Alignment - Calder, Grunwald (1994) (Correct) (32 citations)
these algorithms has been on improving instruction **cache** locality, and the few studies concerned with (ASPLOS-VI)San Jose, California. October 1994. **most** recent address. If the decoded instruction breaks graph so that fall-through branches occur more **frequently**. We use profile information to direct the
www.cs.colorado.edu/~grunwald/GCAG/dirk-arch-aspios94.ps

Profile-Driven Instruction Level Parallel Scheduling with.. - Chekuri Dept (1996) (Correct) (8 citations)
performance in the face of fixed instruction **cache** sizes. In light of this, the threshold and the scheme is computationally intractable in the **most** general case, it is practicable for super **blocks** the branches via hardware support for predicated **execution**, allowing instructions to be moved outside of
theory.stanford.edu/~chekuri/postscript/micro96.ps.gz

Modeling Caching Effect in Continuous Media Server - Kang, Yeom (1999) (Correct)
Modeling **Caching** Effect in Continuous Media Server Sooyong Kang
dcslab.snu.ac.kr/~yeom/paper/mascots99.ps

Practical Issues Of 2-D Parallel Finite Element Analysis - Michelle Hribar (Correct)
machines is the identification of the **most** efficient type of communication scheme for the considerations such as the following are **frequently** neglected: the range of message sizes for which use of parallel processors has made it possible to **execute** large scale applications such as finite element
ece.nwu.edu/pub/CELERO/picpp94.ps.gz

On Caching Search Engine Results - Markatos (1999) (Correct) (7 citations)
On **Caching** Search Engine Results Evangelos P. Markatos
www.ccsf.caltech.edu/~markatos/avg/papers/1999.TR241.Caching_search_engines.ps.gz

Initial Results for Glacial Variable Analysis - Tito Autrey (1996) (Correct) (17 citations)
excellent candidate variables. 5 Related Work The **most** closely related work to glacial variable analysis candidate variables. They are modified much less **frequently** than they are referenced. In current systems the total run-time of the program is reduced. The **execution** time savings must exceed the cost of RTCG.
www.cse.ogi.edu/Sparse/paper/glacial.lcpc.96.ps

Faster Reuse and Maintenance Using "Software Reconnaissance" - Wilde (1994) (Correct) (1 citation)
1994 1. Introduction Just as maintenance is the **most** costly part of the software life cycle, program **Executive** Summary Faster Reuse and Maintenance Using
reuse is that you usually need to understand old **code** in order to make use of it. Probably 30 -40% of
hesperus.oboe.com/serc/TechReports/abstracts/authors/../../files/TR75F.PS

Global Register Allocation Based on Graph Fusion - Guei-Yuan Lueh (1996) (Correct) (7 citations)

compiler to pick the heuristic or strategy that is **most** in line with the rest of the compiler design. 2 compiler design. 2 Background and prior work A **frequently** employed technique is to first allocate a by a Chatin-style allocator. This algorithm uses **execution** probabilities, derived from either profiles or
www.cs.cmu.edu/afs/cs.cmu.edu/project/iwarp/archive/fx-papers/icpc98.ps

Predicting Worst Case Execution Times on a Pipelined RISC.. - Bharrat, Jeffay (1995) (Correct) (3 citations)
Modern computer systems with pipelined processors, **caches**, DMA, etc.can complicate this process. We worst case **execution** times a difficult problem. **Most** programs are written in a higher level language, Predicting Worst Case **Execution** Times on a Pipelined RISC Processor Shaun J.
ftp.cs.unc.edu/pub/users/jeffay/papers/Bharrat.ps.Z

Application-Controlled File Caching Policies - Cao, Felten, Li (1994) (Correct) (52 citations)
Application-Controlled File **Caching** Policies Pei Cao, Edward W. Felten, and Kai Li
Policy The kernel allocation policy is the **most** critical part of two-level replacement. To obtain
ftp.cs.princeton.edu/reports/1994/445.ps.Z

A Hardware Mechanism for Dynamic Extraction and.. - Merten, Trick.. (2000) (Correct) (6 citations)
platform for runtime optimization than trace **caches**, because the traces are longer and persist in Hot Spot Detector [7]At runtime it determines the **most frequently executed** branch instructions while
www.crhc.uiuc.edu/IMPACT/ftp/conference/isca-00-relayout.ps

Efficient Cooperative Caching using Hints - Sarkar, Hartman (1996) (Correct) (31 citations)
Efficient Cooperative **Caching** using Hints Prasenjit Sarkar and John Hartman
www.cs.arizona.edu/swarm/papers/ccache/paper.ps

Some MPEG Decoding Functions on Spert An Example for Assembly.. - Formella (1994) (Correct) (1 citation)
index addresses and all instructions reside in the **cache** we calculate the upper bound of the run time as of Assembly Program 23 1 Introduction The **most** recent documentation (or at least pointer to that operation needs at **most** 4 clock cycle to be **executed**. ffl chaining of operations is possible ffl
ftp.icsi.berkeley.edu/pub/techreports/1994/tr-94-027.ps.gz

The Dark Side of Risk (What your mother never told you about.. - Nicol, Liu (1996) (Correct)
methods employ aggressiveness, but not risk. The **most** widely cited optimistic systems use risk, notably how simulation **code** can be tested to ensure safe **execution** under a risk-free protocol. Whether risky or a parallel discrete-event simulation: a simulation **code** that runs correctly on a serial machine may, when
ftp.cs.dartmouth.edu/TR/TR96-298.ps.Z

Near-Optimal Parallel Prefetching and Caching - Kimbrel, Karlin (1997) (Correct) (28 citations)
Near-optimal parallel prefetching and **caching** Tracy Kimbrel y Anna R. Karlin z August 29,
www.cs.washington.edu/homes/tracyk/focs-long.ps

Efficient Organization of Control Structures in Distributed.. - Hogen, Loogen (Correct)
Nevertheless, their approach yields a good **cache** behaviour, which should also be observed in our In purely sequential implementations of **most** programming languages a runtime stack is used for stacks of several parallel processes, which are **executed** on the same processor element, are stored in an
www-i2.informatik.rwth-aachen.de/OldStaff/hogen/PUBLICATIONS/cc94.ps.gz

First 20 documents Next 20

Try your query at: Amazon Barnes & Noble Google (CiteSeer) Google (Web) CSB DBLP

CiteSeer Find: caching most frequently executed coc Documents Citations

Searching for PHRASE **caching most frequently executed code blocks**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Amazon B&N Google (CiteSeer) Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
1000 documents found. Retrieving documents... **Order: relevance to query.**

DEFLATE Compressed Data Format Specification version 1.3 - Deutsch (1996) (Correct) (4 citations)
as an integer between 0 and 255 does have a **most**- and least-significant bit, and since we write
text usually compresses by a factor of 2.5 to 3 **execut**able files usually compress somewhat less
. 9 3.2.5 Compressed **blocks** (length and distance **codes**) 9 3.2.6
ftp.kiae.su/pub/.1/internet/rfc/rfc1951.ps

Func_mkdb User's Manual - Hol (1988) (Correct)
described is mainly the C programming language. For **most** of the function **blocks** the language only needs to
furthermore changes due to the fact that an **execut**able simulator is created whenever a new function
The function **block** description is translated into C **code**, the C **code** is compiled and the object file is
donau.et.tudelft.nl/pub/space/doc/oldmanuals/func_mkdb.ps.Z

Research on Proof-Carrying Code for Mobile-Code Security - Lee, Necula (1997) (Correct) (3 citations)
a practical approach to mobile-**code** security. The **most** basic obstacle is how to generate the proofs. In
code to be installed dynamically and then **executed**, a host system can provide an flexible means of
Research on Proof-Carrying **Code** for Mobile-**Code** Security A Position Paper Peter
foxnet.cs.cmu.edu/petel/papers/pcc/pcc-mobile.ps

Profile-Guided Context-Sensitive Program Analysis - Debray (Correct) (2 citations)
For example, in the SPEC-95 benchmark m88ksim, the **most frequently** called function, uext(has 19 call
tends to be skewed towards a small number of **frequently executed** call sites. For example, in the
procedure, and back to the basic **block** to which **execut**ion returns at the end of the call. Traditionally,
www.cs.arizona.edu/people/debray/papers/pgcsens.ps

High Performance Celp Coder Utilizing A Novel Adaptive.. - Zijun Yang (Correct)
Lpc **Code**book Sixteenth **Code** Vector Decoded **Block Most** Recently Current **Block** (4) 3) 2) 1) 0) A A A
High Performance Celp **Code**r Utilizing A Novel Adaptive Forward-Backward Lpc
speech signal is re-segmented into overlapping **blocks**. As the LPC coefficients calculated from one of
meru.cecs.missouri.edu/people/vass/adpvq_mmsp_pap.ps.gz

Code Composition as an Implementation Language for Compilers - Stichnoth, Gross (1997) (Correct) (6 citations)
There are many dimensions of quality, but the two **most** critical are correctness and efficiency. While
the complexity of an efficient algorithm to **execute** the statement. Compiling the array assignment
Code Composition as an Implementation Language for
pecan.srv.cs.cmu.edu/afs/cs.cmu.edu/user/stichnot/public/www/dsl97.ps

HARE: A Hierarchical Allocator for Registers in Multiple.. - Berson, Gupta, Soffa (1995) (Correct)
to select values for spilling that will remove the **most** interferences from the graph [BGM 89]
spills inside of nested loops are **executed** more **frequently** than those at a shallower nesting depth or
architectures. HARE makes extensive use of **execut**ion estimates and functional unit availability
www.cs.pitt.edu/~berson/./papers/TR95-06.ps

A Quantitative Analysis of Loop Nest Locality - McKinley, Temam (1996) (Correct) (27 citations)
future directions for architecture and software **cache** optimizations. Since **most** programs spend the
www.masi.uvsq.fr/~temam/Articles/McTe96.ps.gz

Compile/Run-time Support for Threaded MPI Execution on.. - Tang, Shen, Yang (1999) (Correct)
of a permanent variable is small or not aligned to **cache** line size [25, 11]Because of the above
address space and software incompatibility [27]**Most** programs written in MPI, however, should meet our
Compile/Run-time Support for Threaded MPI **Execut**ion on Multiprogrammed Shared Memory Machines
www.cs.ucsb.edu/TRs/techreports/TRCS98-30.ps

Optimizing ML with Run-Time Code Generation - Leone, Lee (1995) (Correct) (91 citations)
it involves some subtle interactions with the **cache**-memory system and the instruction pre-fetching
and abstraction are desirable design goals for **most** software systems. But, in practice, the costs of

for run-time **code** generation because it is **frequently** the innermost loop of long-running numerical
foxnet.cs.cmu.edu/~petel/papers/staged/mleone-pldi96.ps

The Effect of Client Caching on File Server Workloads - Kevin Froese (1996) (Correct) (6 citations)
The Effect of Client **Caching** on File Server Workloads Kevin W. Froese
www.cs.usask.ca/staff/kwf230/research/hicss96.ps.gz

Experience with Automatic Mapping of Sensor--Based Applications - Jaspal Subhlok (1995) (Correct) (1 citation)
under these conditions later in this section. **Cache** effects An important change that occurs when
problem in parallel computing is to find the **most** efficient mapping of a parallel program onto the
computing, all available processors combine to **execute** every computation step in a program. Complex
www.cs.cmu.edu/~jass/papers/pdpta97.ps

Lightweight Run-Time Code Generation - Leone, Lee (1994) (Correct) (34 citations)
template compilation include decompression and **cache** simulation [KEH93] and the bitblt graphics
Static analyses are inherently imprecise because **most** interesting aspects of run-time behavior are
3 and applying aggressive optimizations only to **frequently executed** methods using dynamic recompilation.
www.cs.cmu.edu/afs/cs.cmu.edu/user/mleone/papers/lw-rtcg.ps

Iteration Abstraction in Sather - Stephan Murer (1996) (Correct) (2 citations)
requires the explicit implementation of suitable **caching** heuristics. Many other classes similarly define
loop .end statement. While this suffices for the **most** basic iterative tasks, we felt the need for a more
in our experience such bugs have not arisen **frequently** in practice, although this may not hold true
www.fit.qut.edu.au/~szypersk/pub/TOPLAS96.ps.gz

A Case for Delay-Conscious Caching of Web Documents - Scheuermann, Shim, Vingralek (1997) (Correct) (21 citations)

A Case for Delay-Conscious **Caching** of Web Documents Peter Scheuermann*Junho
www.bell-labs.com/user/rvingral/www97.ps

Binary Translation: Static, Dynamic, Retargetable? - Cifuentes, Malhotra (1996) (Correct)
machine instruction [23] in the 1990s by using **caching** techniques. Binary translation is still a young
of software is a considerable investment by **most** organizations. A US survey in the late 1970s
a runtime environment to successfully support the **execution** of the translated programs on the new machine.
www.it.uq.edu.au/personal/cristina/icsm96.ps

Towards a Better Understanding of Web Resources and Server... - Wills, Mikhailov (1999) (Correct) (20 citations)
of Web Resources and Server Responses for Improved **Caching** Craig E. Wills and Mikhail Mikhailov Computer
www.cs.wpi.edu/~mikhail/papers/www8.ps.gz

Continuous Multicast Push of Web Documents over the Internet - Rodriguez, Biersack (1998) (Correct) (3 citations)
that change very **frequently** and that are not worth **caching**. A Web server using CMP continuously multicasts
[1] 3] 6]where a Web server pushes the **most** recent version of a document to a group of
We propose the distribution of very popular and **frequently** changing Web documents using continuous
www.eurecom.fr/~btroup/BPublished/RODRI98_cmp.ps.gz

Building Interpreters by Composing Monads - Steele, Jr. (1994) (Correct) (47 citations)
Figure 15. The continuation building **block** was the **most** difficult to construct-it took a long time to
Abstract: We exhibit a set of functions **coded** in Haskell that can be used as building **blocks** to
coded in Haskell that can be used as building **blocks** to construct a variety of interpreters for
www-swiss.ai.mit.edu/ftpdir/users/dae/related-papers/steele.ps.Z

Using Profile Information to Assist Classic Code Optimizations - Chang (1991) (Correct) (53 citations)
W. Hwu and P. P. Chang, Achieving High Instruction **Cache** Performance with an Optimizing Compiler"
profiling tools allow programmers to identify the **most** important functions and the **most frequently**
the **execution** time by moving instructions from **frequently executed** program regions to infrequently
ftp.crhc.uiuc.edu/pub/IMPACT/journal/spe.profile-classic.91.ps

Documents 21 to 40 Previous 20 Next 20

Try your query at: Amazon Barnes & Noble Google (CiteSeer) Google (Web) CSB DBLP